Practical 12

AIM: - Implement echo client server using TCP/UDP sockets.

Algorithm:-

Server Side:

1. Start the program.
2. Create a socket using socket() function with TCP (AF_INET, SOCK_STREAM).
3. Bind the socket to an IP address and port number using bind().
4. Listen for incoming connections using listen().
5. Accept a client connection using accept().
6. Receive data from the client using recv().
7. Send the same data back to the client using send() (echoing).
8. Close the client connection.
9. Stop the server program.

Client Side:

1. Start the program.
2. Create a socket using socket() with TCP (AF_INET, SOCK_STREAM).
3. Connect to the server using connect().
4. Input a message from the user.
5. Send the message to the server using send().
6. Receive the echoed message from the server using recv().
7. Display the echoed message on the screen.
8. Close the connection.
9. Stop the client program.

Tcp server(server.py)

```
import socket
s = socket.socket()
s.bind(('localhost', 12345))
s.listen(1)
print("Server waiting for connection...")
conn, addr = s.accept()
print("Connected with", addr)
data = conn.recv(1024).decode()
print("Received from client:", data)
conn.send(data.encode())
conn.close()
s.close()
```

Tcp client(client.py)

```
import socket
s = socket.socket()
s.connect(('localhost', 12345))
msg = input("Enter message: ")
s.send(msg.encode())
data = s.recv(1024).decode()
print("Echoed from server:", data)
s.close()
```

Client input:

```
Enter message: Hello Server
```

Server output:

```
Server waiting for connection...
Connected with ('127.0.0.1', 56732)
Received from client: Hello Server
```

Client output:

```
Echoed from server: Hello Server
```